AMENDMENTS TO THE SPECIFICATION

Applicant presents replacement paragraphs below indicating the changes with insertions indicated by underlining and deletions indicated by strikeouts and/or double bracketing.

Please replace the paragraph/line beginning at page **4**, line **11** with the amended paragraph/line as follows:

Fig. 6A-6[[B]]C are flowcharts describing a class room login process;

Please replace the paragraph beginning at page **4**, line **31** with the amended paragraph as follows:

Computer system 101 may be a general purpose computer system that is programmable using a high level computer programming language. The computer system may also be implemented using specially programmed, special purpose hardware. In the computer system 101, the processor 105 is typically a commercially available processor, such as the PENTIUM® microprocessor from the Intel Corporation, PowerPC microprocessor, SPARC processor available from Sun Microsystems, or 68000 series microprocessor available from Motorola. Many other processors are available. Such a processor usually executes an operating system which may be, for example, the DOS, WINDOWS 95, or WINDOWS NT available from the Microsoft Corporation, MAC OS SYSTEM 7 available from Apple Computer, SOLARIS available from Sun microsystems, NetWare available from Novell Incorporated, or UNIX available from various sources.

Please replace the paragraph beginning at page **5**, line **9** with the amended paragraph as follows:

The communication network 104 may be an ETHERNET network or other type of local or wide area network (LAN or WAN), a point-to-point network provided by telephone services, or other type of communication network. Information consumers and providers, referred to as client [[103]]102 and server [[102]]103 systems, respectively, communicate through the network 104 to exchange information. Computer system 101 may be configured to perform as a client 102 or server 103 system or both on the network 104. A server 103A may store structured documents on a storage device 109 located on the server 103A. The server may also provide

these structured documents to one or more client systems 102 in response to a request generated by a client 102A through the network 111. Similarly, structured documents may be created, edited, viewed, or converted on such client 102 and server 103 systems.

Please replace the paragraph beginning at page **11**, line **9** with the amended paragraph as follows:

The server model object includes a number of methods associated with it. There is a method referred to as an "onAttachedEvent" method that indicates when a new client has attached to the model. Specifically, the server model is notified of a client attach event after the server sends a client attach event to other clients. [[As]]An "onSendEvent" method handles an event sent by a corresponding client model. Similarly, there are "onSendRealTimeEvent" and "onSendBinaryEvent" methods for handling real time and binary events, respectively.

Please replace the paragraph beginning at page **11**, line **19** with the amended paragraph as follows:

View 407 and controller 408 [[controller]] objects may differ substantially between models with which they are associated. The controller handles all events from the view that are reflected state changes in the model. In controller has a reference to a model in which the controller can call model methods. An additional controller variable may be used to indicate whether the controller is enabled or disabled.

Please replace the paragraph beginning at page **13**, line **23** with the amended paragraph as follows:

Text information within an event may be, for example, a string representing a URL, or location to a resource on the network. The object server [[object]] process [[10]]210 may provide resource locations to a client browser program 203 for loading multimedia data. The events generated by the server 103 may be used to update a local client model, that is, a resource location string may be sent to the client 102 to direct the browser program 203 to load an updated multimedia resource.

Please replace the paragraph beginning at page **15**, line **1** with the amended paragraph as follows:

Before beginning a course, a student registers for a course, and the browser program 203 provides applets 405 for performing course functions. The student "browses" to a content server on a client 102 using a starting location URL for the course. The content server then returns a starting structured document which contains links for installing the applets 405 and for accessing the course catalog 401. The student first installs the applets [[404]]405, which may be downloaded by an installer applet and running the installation applet.

Please replace the paragraph beginning at page **16**, line **1** with the amended paragraph as follows:

The process 600 for logging into the course will now be described. As shown in Fig. 6A, the process 600 begins at step 602. At step 604, the student selects a course link on the [[student 's]] student's personal home page 403. The browser program 203 sends the link to the content server process 208 at step 606. The link includes a reference to the registration object 409 for the course. At step 608, the content server 208 runs a JAVA servlet and passes the servlet the registration ID for the student, which is a unique value. The servlet uses the registration ID to look up the registration object 409 in the database containing the student and course information at step 610. At step 612, the servlet then generates an HTML frame document that defines the user interface layout of a browser program view. This HTML frame document is based on an HTML template and contains the student's ID and the course number. The frame document defines primary frames: the control panel frame, the participant frame, and the media frame described below and as shown in Figure 9B. The control panel and participant frames are initially left blank. The media frame contains a URL to a Loader HTML document. The Loader HTML document is a template document located on the server that gets the password from the student and "loads" an associated "Loader" applet.

Please replace the paragraph beginning at page **16**, line **17** with the amended paragraph as follows:

The Loader HTML document contains the student's ID and requests an optional course password at step 614. The student enters the password at step 616 and selects "Login" from the browser program 203 interface. The browser program 203 sends the login data to the content

server process 208 to [[the]] a servlet that verifies the section password and returns a generated HTML document containing the Loader applet at step 618. The Loader applet is run in the student's browser program 203 and checks to make sure the correct applets 405 have been installed (showing an error message if it has not), preloads JAVA class libraries and controls the browser program 203 to display two HTML documents in the control panel and participant frames, respectively. At step 620, these HTML documents cause the content server to run JAVA servlets to generate the HTML documents that contain the JAVA applets 405. These pages are generated from templates and one of them, the participant document contains the registration object ID and the conference ID as a parameters to a "people" applet. The "people" applet is the master applet, that it, an applet that monitors the status of other applets. The conference ID references a conference object in the database that uniquely identifies the section's conference.

Please replace the paragraph beginning at page **17**, line **7** with the amended paragraph as follows:

As each applet 405 is executed by the browser program 203, each applet registers itself with the master applet and initializes itself at step [[625]]624. Applets, when executed, instantiate objects. The objects define methods, such as an "init" method, which defines variables and other settings for the object. Final initialization is completed when all applets 405 have registered and are ready to accept data. As discussed above, one special applet, the people applet, performs the function of the master applet. The people applet makes the original connection request to the object server process 210. This request contains the conference ID for the course. The object server process 210 determines if the course associated with the conference ID is already created at step 626 and, if it is determined at step 628 that the conference object is not created, the object server 210 creates the conference object and the default "main" room object at step 630. An associated student conference and the default "main" room it contains are created by the framework.

Please replace the paragraph beginning at page **18**, line **6** with the amended paragraph as follows:

To start a tool, a student selects a tool by selecting an icon on the control panel in a browser program window at step 704 (A view of the browser program window is shown in Fig.

9B). The client browser loads applet code for the tool that is selected, and, in step 706 executes the applet with input parameters from an HTML document referenced by the control panel tool icon. The applet, when executed, instantiates the client model object at step 708. At step 710, the client model object contacts the object server process 210 with an attach event. If, at step 712, the server model does not exist, the applet instantiates the server model at step 714. If the server model does exist, the client sets the client model state to the server model state at step 716. At step 718, the object instantiation process 700 is complete, and the client model object waits for update events from the server model object.

Please replace the paragraph beginning at page **18**, line **27** with the amended paragraph as follows:

In Figure 8B, a controller object on the client may provide an update to the server model object, and the change in state of the server model object may be propagated to client model objects. At step 814, the local controller object 408 listens for input from a user to the controller object 408. If, at step 816, a user input provided an update of the local model object 406, an update event is sent to a model object 411 on the server 103. At step 820, the object server process 210 broadcasts an update event to clients 102. At step 822, clients 102 update their client model objects 406 according to the update event, and at step 824, clients 102 update their client view objects 407 according to the update event.

Please replace the paragraph beginning at page **19**, line **1** with the amended paragraph as follows:

In Fig. 8C, the client may update an associated client model object at step 822 of Fig. 8B according to the update process 830. At step 832, process 830 begins. In step 834, [[The]]the model object processes an event message received from server 103. The event message may contain a string which corresponds to a URL of an updated multimedia resource on server 103. The URL string may be provided by a method within the model object to the browser program 203 at step 836. At step 838, the browser program loads the multimedia resource from the content server process 208. As discussed above, the multimedia resource may be an HTML document containing updated content. At step 840, the client model update process 830 ends.

Please replace the paragraph beginning at page **19**, line **10** with the amended paragraph as follows:

Figure 9A shows a graphical user interface of a design application referred to as a course editor 900 used to create multimedia content. The multimedia content may be added to a course by running course editor applets running within a browser program 203. Figure 9A shows a program browser view 901 having a design window 902 through which multimedia content is viewed. The multimedia content could include HTML documents, slides, images, or any multimedia information that may be displayed through a browser program 203. The directory frame 903 shows the sequence of multimedia information to be presented in the course. The browser program view 901 that runs the course builder applet associates course content with a URL 904. This URL 904 may be provided to a client browser program 203 to display in a client view 910 of the course.